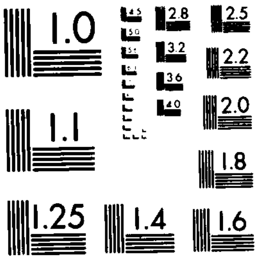
AD-A178 564 JCL (JOB CONTROL LANGUAGE) PROCEDURES TO RUN THE HULL 1/1
CODE ON THE CYBER 2..(U) MATERIALS RESEARCH LABS ASCOT
VALE (AUSTRALIA) D L SMITH NOV 86 MRL-TN-500
UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
5 87



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS 1963-A

MRL-TN-508 AR-004-849

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

MATERIALS RESEARCH LABORATORIES

MELBOURNE, VICTORIA

TECHNICAL NOTE

MRL-TN-508

JCL PROCEDURES TO RUN THE HULL CODE ON THE CYBER 205 COMPUTER INSTALLED ON CSIRONET

David L. Smith

Approved for Public Release

DEPARTMENT OF DEFENCE
MATERIALS RESEARCH LABORATORIES

TECHNICAL NOTE

MRL-TN-508

JCL PROCEDURES TO RUN THE HULL CODE ON THE CYBER 205 COMPUTER INSTALLED ON CSIRONET

David L. Smith

ABSTRACT

Version #120 of the HULL computer code is being implemented at MRL. Two computers are used: a Cyber 205 supercomputer, and a Cyber 845 front-end, both hosts on the CSIRONET network. This report explains a complex set of JCL procedures developed to run HULL on this installation.

Approved for Public Release

POSTAL ADDRESS: Director, Materials Research Laboratories
P.O. Box 50, Ascot Vale, Victoria 3032, Australia

SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED

DOCUMENT CONTROL DATA SHEET

REPORT NO.	AR NO.	REPORT SECURITY CLASSIFICATION
MRL-TN-508	AR-004-849	Unclassified

TITLE

JCL procedures to run the hull code on
the cyber 205 computer installed on CSIRONET

AUTHOR(S)	CORPORATE AUTHOR
David L. Smith	Materials Research Laboratories PO Box 50, Ascot Vale, Victoria 3032

REPORT DATE	TASK NO.	SPONSOR
November 1986	DST 86/250	DSTO
FILE NO.	REFERENCES	PAGES
G6/4/8-3293	4	22

CLASSIFICATION/LIMITATION REVIEW DATE	CLASSIFICATION/RELEASE AUTHORITY
	Superintendent, MRL Physical Chemistry Division

SECONDARY DISTRIBUTION

Approved for Public Release

ANNOUNCEMENT

Announcement of this report is unlimited

KEYWORDS

Hull Hydrodynamic code	Computer programs

COSATI GROUPS 0902

ABSTRACT

Version #120 of the HULL computer code is being implemented at MRL. Two computers are used: a Cyber 205 supercomputer, and a Cyber 845 front-end, both hosts on the CSIRONET network. This report explains a complex set of JCL procedures developed to run HULL on this installation.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

CONTENTS

JCL PROCEDURES TO RUN THE HULL CODE ON THE CYBER 205 COMPUTER INSTALLED ON CSIRONET

1. INTRODUCTION

The HULL code [1] is a suite of programs for solving problems of hydrodynamic flow. HULL is actually a self contained system, with internal maintenance and editing facilities, currently capable of running on a number of computer sites throughout the world. A utility for the HULL system called SAIL [2] manages the internally coded system, and is used to generate Fortran programs appropriate to the problem being solved.

The Materials Research Laboratories (MRL) obtained a current version of the HULL system early in 1986 from Eglin AFB, through TTCP channels. This version (HULL 120) is capable of solving problems in two or three dimensions, using Eulerian or Lagrangian equations of motion, or a combination of both. HULL 120 has models for elastic/plastic material behaviour, and explosive burn routines, and will be used at MRL principally to study explosives-related phenomena. (An earlier version, HULL 20, has been in use at MRL for some years, but this has fewer capabilities, and is only used for airblast studies).

HULL 120 is currently being installed and tested on a Control Data Cyber 205 supercomputer on the CSIRONET network. Part of the installation involves embedding into the HULL system the necessary parameters and changes to make HULL-generated code run correctly on the 205. This process is not yet complete, and will be reported separately. A second part of the installation involves writing the job control language (JCL) to run HULL. This report describes a complex set of JCL procedures developed at MRL to run HULL programs on the CSIRONET Cyber 205.

2. DISCUSSION

2.1 Using the HULL system

HULL 120 is maintained by a utility called SAIL, which operates on coded identifiers and directives embedded in the HULL library. SAIL is similar to a portable version of Control Data's UPDATE utility [3], but has extended capabilities such as internal arithmetic and logic processing for calculation of internal parameters.

The HULL system is large: over 200,000 records, approximately half of which are lines of Fortran code. It is not simple to use, as several steps are needed to extract an ordinary Fortran program. Sections 2.2 to 2.4 explain program generation in more detail, with the aid of flowcharts. Table 1 contains a glossary of the terms used in describing HULL 120 and its installation.

Once HULL has been installed, the user will normally generate and run three major program segments. KEEL is the pre-processor which defines and fills the computational grid. HULL (the name of one of the programs as well as the whole system) is the cycler which solves the difference equations and moves the problem through time. PULL is the post-processor which generates graphical output from the HULL calculations. The JCL procedures described in this report are designed to help generate and run KEEL, HULL and PULL.

2.2 Installation on CSIRONET

The Cyber 205 computer on the CSIRONET network is available as a remote host, accessible from a number of computers via a Control Data "Loosely Coupled Network". HULL 120 has been installed using a Cyber 845 running the NOS operating system to access the 205 running the VSOS operating system.

Because the 205 can only be accessed in batch mode, and because editing facilities and system utilities on VSOS are rather poor, the SAIL utility and the HULL library are maintained on the 845. Figure 1 shows a flow chart for the initial installation of SAIL and HULL. SAIL was modified to run on NOS, and then used to generate a packed HULL library from the tape file. SAIL was then used in conjunction with the HULL library to generate two files used in most HULL applications. The first, a program called PLANK, is used to generate expanded secondary input from the normal input written by the HULL user. This technique saves the user from having to create complex inputs by hand. PLANK is normally generated only once for any given set of installation parameters (eg computer word size, character bit length etc). The second file, MATLIB, contains the material property constants for all materials used in HULL calculations. This library file will also not need regeneration unless new materials are added to the system.

TABLE 1

Glossary of terms used in describing HULL 120 and its installation

HULL	Program to solve the difference equations and time iterations (cycler). Also, loosely, the HULL 120 system.
HULLJCL	NOS procedure to assemble JCL to run KEEL, HULL or PULL.
HULLJOB	File containing JCL generated by HULLJCL
HULL LIBRARY	File containing entire HULL 120 system in packed form.
HULL 120	Version #120 of the entire HULL system.
KEEL	Program to define and fill the computational grid (pre-processor).
MATLIB	File containing material property constants.
PLANK	Program to expand user input and generate secondary input for SAIL.
PRIMARY INPUT	Input for KEEL, HULL or PULL generated by user.
PULL	Program to process HULL results and generate graphical output (post-processor).
SAIL	Utility to manage HULL 120, operating on coded internal identifiers and directives.
SECONDARY INPUT	Input for SAIL to generate appropriate version of KEEL, HULL or PULL from HULL LIBRARY.
USER INPUT	See PRIMARY INPUT.

2.3 Job Flow and JCL Design

The aim has been to write a set of JCL commands that make it easier for the user to generate and run the KEEL, HULL and PULL programs. As both 205 and 845 jobs are submitted from the 845, a NOS "procedure" [4] has been used to construct the JCL. Procedures allow interactive entry of input parameters, and dynamic substitution of these parameters into the procedure body. Procedures also allow IF blocks for conditional skipping of commands. This has made it simple to input account names, identifiers, time limits etc, and guarantee error free multiple insertions into the JCL.

Figure 2 is a flow chart of a job to run one of the main HULL programs. The job spends two phases on the 845 and two phases on the 205.

Phase 1 is the startup on the 845, which passes the input file to the 205, and submits the remaining JCL to the 205. (This is the only method available for running interdependent jobs between the two hosts, and a resubmission of remaining JCL is done between each phase). Phase 2 is a PLANK run on the 205, where the user input is analysed and a secondary input generated ready for SAIL. Phase 3 is a SAIL run on the 845, where SAIL uses the secondary input parameters to generate a main program (KEEL, HULL or PULL) from the packed HULL library. Phase 4 is the run of this program on the 205.

During the job, a log file of dayfiles and minor outputs is maintained on the 845, and appended to at the end of each phase. Files are managed in such a way that intermediate files are kept in the event of error exits, but deleted once a phase of the job has been successfully completed. File housekeeping is achieved by using a problem identifier as part of the filename for key files, and purging old copies if they already exist from a previous run.

Appendix A contains a listing of the NOS procedure HULLJCL which assembles the JCL to run a job as shown in Figure 2.

2.4 Using the HULLJCL Procedure on NOS

User input to the HULLJCL procedure is interactive, and prompts are included to help the user. Input parameters are as follows:

NUID	NOS user ID
NUPW	NOS user password
NPID	NOS project ID
GROUP	NOS group charge code
VUID	VSOS user ID
VUPW	VSOS user password
VPID	VSOS project ID
PROG	Name of program to be run
TIMELIMIT	Time limit for main VSOS run
PROBLEMID	4 character problem identifier
DATEFILE	Name of file containing HULL input data
DEFERNOS	Run NOS jobs at deferred rate (yes/no)

When HULLJCL has been run with these inputs, a local file HULLJOB is created which contains the JCL to run one of the main programs, KEEL, HULL or PULL. HULLJOB can then be submitted to the batch queues. Appendix B shows a typical HULLJCL input and resultant HULLJOB output.

For complete solution of a problem using HULL 120, all three main programs will be run. Files passed between the programs are automatically kept and attached by the procedure. Figure 3 shows the organisation of files used and passed by KEEL, HULL and PULL. In some cases, minor manual adjustments to HULLJOB may be necessary for file assignments in a PULL run, as PULL has many I/O possibilities. The procedure issues a warning to this effect whenever the PULL program is being generated.

3. CONCLUSIONS

Running the HULL 120 system on CSIRONET requires complicated jobs and file handling, and hence verbose JCL. The NOS procedure HULLJCL offers a convenient way of assembling the JCL, and minimises the risk of errors in job construction.

4. ACKNOWLEDGEMENTS

The author would like to thank Dr Robert Bell of CSIRO Division of Atmospheric Research and David Micklethwaite of Control Data Australia for their advice on the JCL, and John Waschl of MRL for his advice on running HULL 120.

Grateful acknowledgement is also made to Eglin AFB for providing HULL 120.

5. REFERENCES

1. Matuska, D.A. and Osborn, J.J. "HULL Documentation", Orlando Technology Inc., Florida, 1985.

2. "Sail User's Guide", Shalimar Research and Technology Inc., Florida, 1983.

3. Update Reference Manual, Control Data Corporation, Manual 60449900. (Update is a utility for maintaining large files).

4. Chapter 4, NOS Version 2 Reference Set, Control Data Corporation, Manual 60459680.

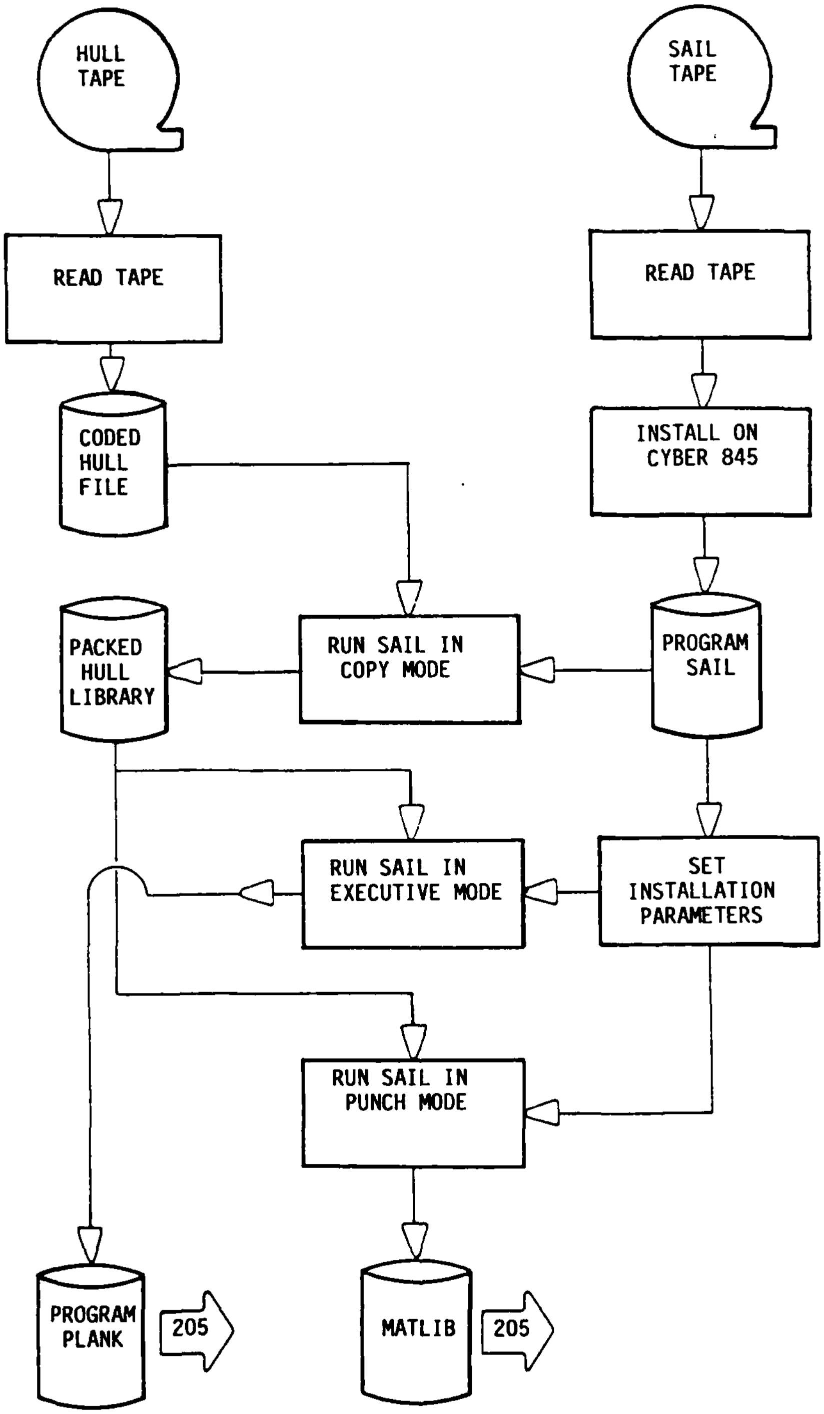


FIGURE 1 Initial Installation of HULL on a Cyber 845

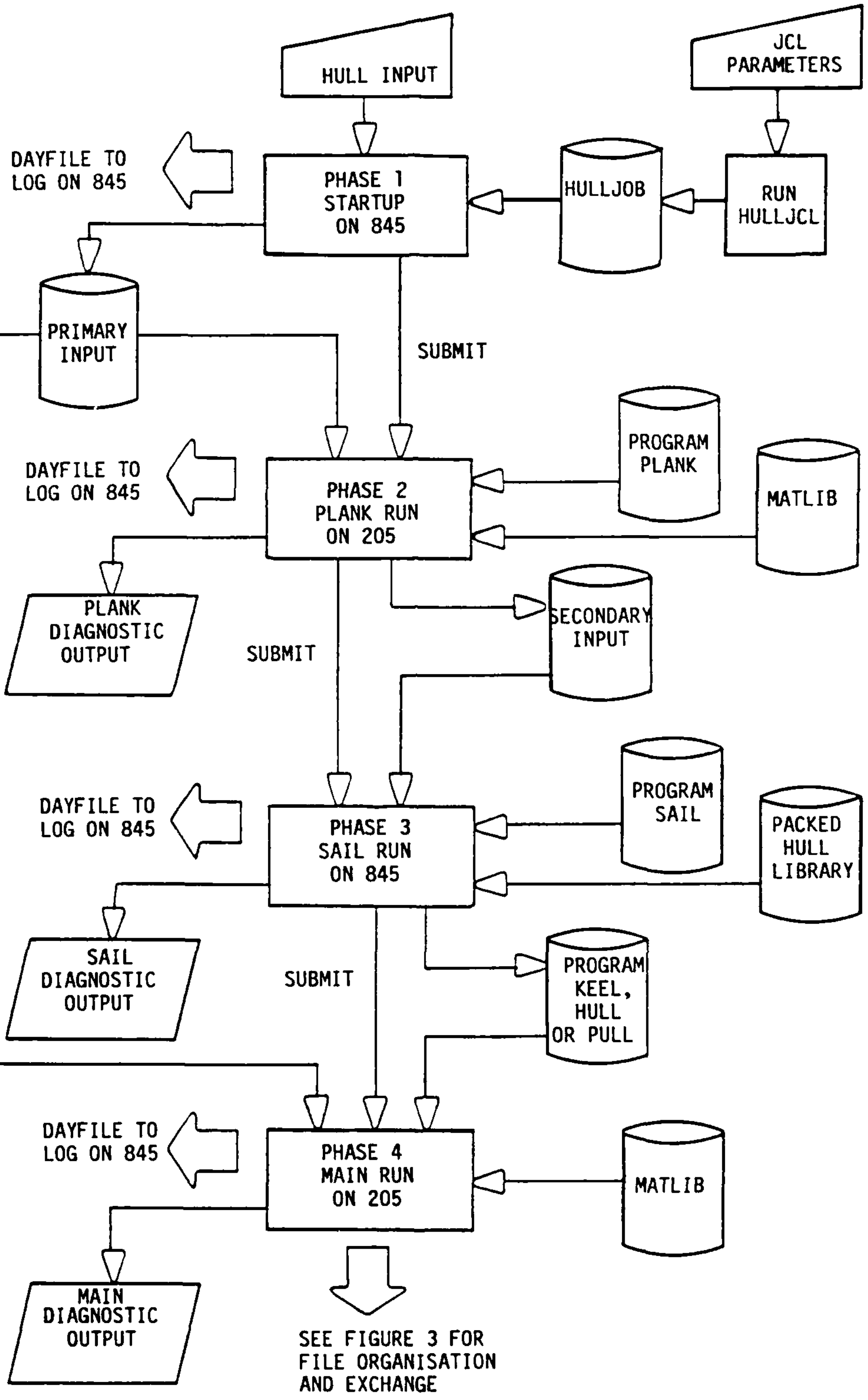


FIGURE 2 Flowchart of a job to generate and run one of the main HULL programs

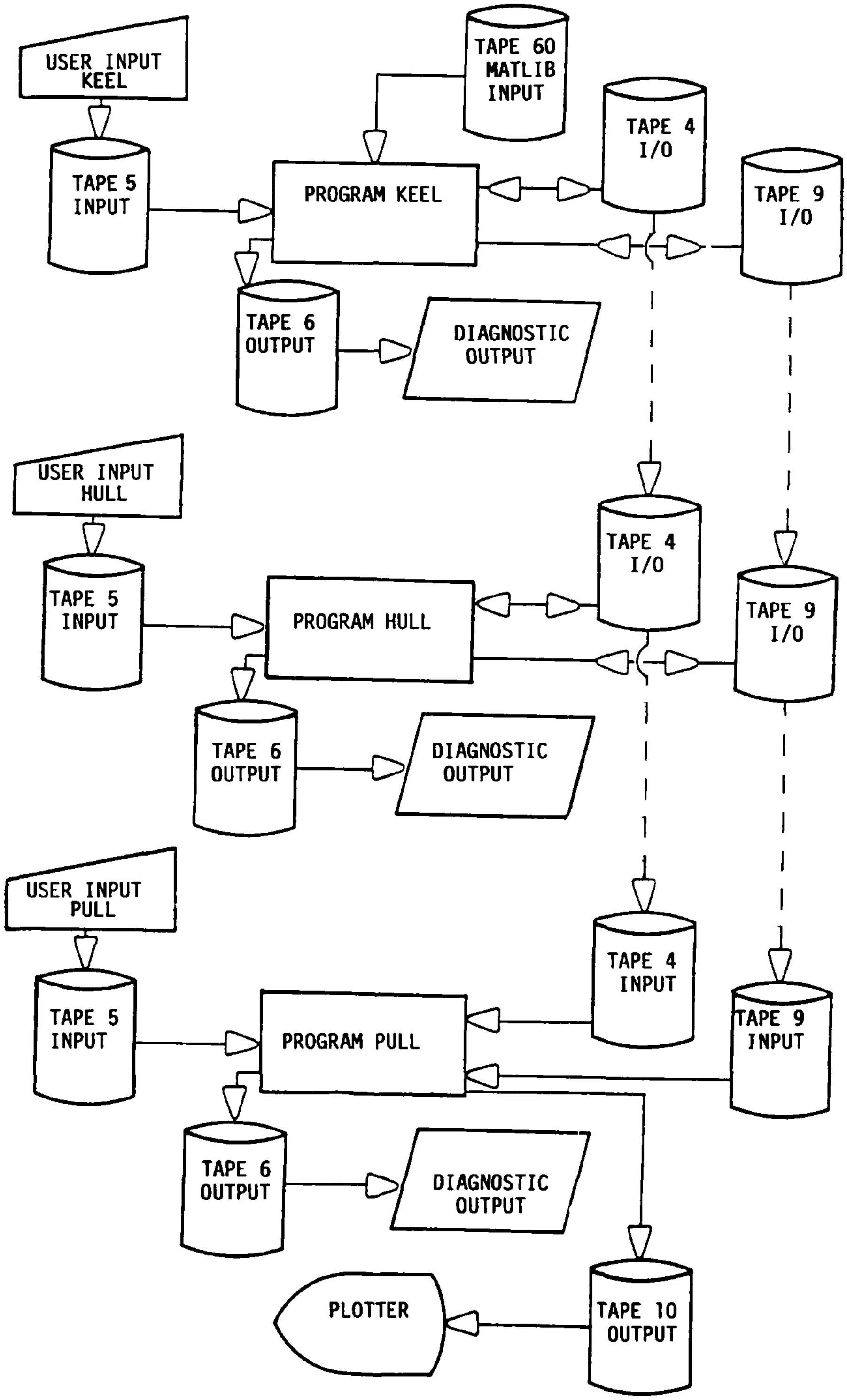


FIGURE 3 File organisation and exchange for main HULL programs

APPENDIX A

LISTING OF THE NOS PROCEDURE HULLJCL

```
.PROC,HULLJCL*I" Proc to set up a NOS/VSOS job to run a HULL program",
NUID'NOS user name (UID)',
NUPW'NOS user password',
NPID'NOS project id (PID)',
GROUP'NOS charge number (First three characters of your PID)',
VUID'VSOS user name or number (UID), LF if same as NOS'=(*F,$$=SAME),
VUPW'VSOS user password, LF if same as NOS'=(*F,$$=SAME),
VPID'VSOS project id (PID), LF if same as NOS'=(*F,$$=SAME),
PROG'Name of HULL program to be run'=(KEEL=KEE,HULL=HUL,PULL=PUL),
TIMELIMIT'Time limit for main VSOS run'=(*S4(1234567890)),
PROBLEMID'Problem identifier up to 4 alphanumeric characters'=
(*S4(ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890)),
DATFILE'Name of NOS permanent file containing input data'=(*F),
DEFERNOS'Run the NOS jobs deferred'=(Y=DEF,YES=DEF,N=CYN,NO=CYN).
.HELP
      Procedure to assemble JCL for running a program from the HULL suite.
      The job is written as local file HULLJOB, but not submitted.

      To submit, type SUBMIT,HULLJOB,TO.

                  ***Author David L. Smith,  July 1986.***
                                 *********
.ENDHELP
.IF,$PROG$.EQ.$PUL$,WARNING.
NOTE(OUTPUT,NR)* WARNING - PULL FILE ASSIGNMENTS MAY NEED ADJUSTMENT.
NOTE(OUTPUT,NR)* CHECK THE PHASE 4 VSOS RUN DEPENDING ON TYPE OF PULL RUN.
.ENDIF,WARNING.
NOTE(OUTPUT,NR)* JOB SAVED AS LOCAL FILE HULLJOB.....
.DATA,HULLJOB
.*        **************************
.*        * BEGIN PHASE 1 ON NOS *
.*        **************************
SUB_PROG_L,ST_DEFERNOS.  <<<SEND THE DATA TO VSOS AND START THE RUN>>>
USER,NUID,NUPW.
CHARGE,GROUP,NPID.
SETJOB,DC=NO.
COMMENT.  GET CR ATTACH THE INPUT DATA TO GO TO VSOS.
GET,INDATA=DATFILE/NA.
IFE,.NOT.FILE(INDATA,AS),DOATT.
ATTACH,INDATA=DATFILE.
ENDIF,DOATT.
COMMENT.  SEND THE DATA FILE TO VSOS VIA MFLINK
MFLINK,INDATA,ST=CYV,DD=C6.
.IF,$VUID$=$SAME$.OR.$VUPW$=$SAME$.OR.$VPID$=$SAME$,SAMEPID.
*USER,U=NUID,AC=NPID,PA=NUPW.
.ELSE,SAMEPID.
*USER,U=VUID,AC=VPID,PA=VUPW.
.ENDIF,SAMEPID.
*PURGE,DAT_PROBLEMID.
*MFTAKE,DAT_PROBLEMID.
*DEFINE,DAT_PROBLEMID.
COMMENT.  SEND THE REMAINING JOBS TO VSOS VIA MFQUEUE
```

```
COPYCR,INPUT,VJOB1,3.
MFQUEUE,VJOB1,ST=CYV,DC=IN.
ENQUIRE,SR.
COMMENT.  START THE LOG FILE ON NOS
PURGE,LOG_PROBLEMID/NA.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
SAVE,DF=LOG_PROBLEMID.
.IF,$VUID$.NE.$SAME$,PERMIT.
PERMIT,LOG_PROBLEMID,VUID=A.
.ENDIF,PERMIT.
EXIT.
COMMENT.  RETRIEVE DAYFILE AND OUTPUT ON ABORT
ENQUIRE,SR.
PURGE,LOG_PROBLEMID/NA.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
SAVE,DF=LOG_PROBLEMID.
.EOR
.*        *************************
.*        * BEGIN PHASE 2 ON VSOS *
.*        *************************
PLANKRUN,STCYV.  <<<RUN PLANK ON VSOS TO GENERATE INPUT2>>>
.IF,$VUID$=$SAME$.OR.$VUPW$=$SAME$.OR.$VPID$=$SAME$,SAMEPID.
USER,U=NUID,AC=NPID,PA=NUPW.
.ELSE,SAMEPID.
USER,U=VUID,AC=VPID,PA=VUPW.
.ENDIF,SAMEPID.
RESOURCE,JCAT=DEFERRED,TL=300,LP=14,WS=1792.
SET,DAYFILE=OFF.
PATTACH,HULLPOOL.
ATTACH,DAT_PROBLEMID.
COPY,DAT_PROBLEMID,TAPE5.
.IF,$PROG$.EQ.$HUL$,KFILES.
ATTACH,KEE4_PROBLEMID,AC=WRAXM.
COPY,KEE4_PROBLEMID,TAPE4.
.ENDIF,KFILES.
.IF,$PROG$.EQ.$PUL$,HFILES.
ATTACH,HUL4_PROBLEMID,AC=WRAXM.
COPY,HUL4_PROBLEMID,TAPE4.
.ENDIF,HFILES.
FTN200,I=PLANK121,B=PLANKBIN/500,SC=1,LIST=0,C64,OPT=0,ERRORS=ERRLIST.
LOAD,PLANKBIN,CN=PLANKEXE/5000,CDF=5000,L=NULL.
PURGE,TAPE6.
DEFINE,TAPE6.
PLANKEXE.
REWIND,TAPE6.
COMMENT.  SEND PLANK OUTPUT TO NOS
```

```
COPY,TAPE6,T6TEMP.
MFLINK(T6TEMP,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
  "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
   "PURGE,PLNKOUT/NA.","SAVE,PLNKOUT/NA.")
COMMENT.  SEND SECONDARY INPUT FILE TO NOS
MFLINK(INPUT2,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
  "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
   "PURGE,IN2_PROBLEMID/NA.","SAVE,IN2_PROBLEMID.")
COMMENT.  SEND REMAINING JOBS TO NOS VIA MFQUEUE
COPY,INPUT,SAILJOB.
COPYL,INPUT,SAILJOB,PART=G,NUM=1,OSKIP=*.
MFQUEUE,SAILJOB,ST=CYN,DD=C6,DC=IX.
PURGE,TAPE6.
SUMMARY.
COMMENT.  APPEND DAYFILE ETC TO LOG FILE ON NOS
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
  "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
  "APPEND,LOG_PROBLEMID/UN=NUID,NA.".
EXIT.
COMMENT.  RETRIEVE DAYFILE ETC ON ABORT
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
  "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
  "APPEND,LOG_PROBLEMID/UN=NUID,NA.".
.EOR
.*        ************************
.*        * BEGIN PHASE 3 ON NOS *
.*         **************************
SAILRUN,ST_DEFERNOS.  <<<RUN SAIL ON NOS TO GENERATE VERSION OF PROG_L>>>
USER,NUID,NUPW.
CHARGE,GROUP,NPID.
SETTL,200.
SETJOB,DC=NO.
COMMENT. RUN SAIL ON NOS TO GENERATE THE MAIN PROGRAM.
PURGE,SAILOUT/NA.
PURGE,PROG_PROBLEMID/NA.
DEFINE,PROG_PROBLEMID.
ATTACH,HVER121/UN=DFCTJW,PN=RCYN002.
ATTACH,BINLIB/UN=DFCDLS,PN=RCYN002.
GTR,BINLIB,LGO.REL/*
GET,IN2_PROBLEMID.
COPYCR,INPUT,PROG_L_JOB,1.
REWIND,*.
LGO,INPUT2=IN2_PROBLEMID,OUTPUT=SAILOUT,OLD=HVER121,SOURCE=PROG_PROBLEMID.
SAVE,SAILOUT/NA.
COMMENT.  SEND THE PROGRAM TO VSOS.
REWIND,*.
COPY,PROG_PROBLEMID,LOCALF.
MFLINK,LOCALF,ST=CYV,DD=C6.
.IF,$VUID$=$SAME$.OR.$VUPW$=$SAME$.OR.$VPID$=$SAME$,SAMEPID.
*USER,U=NUID,AC=NPID,PA=NUPW.
.ELSE,SAMEPID.
*USER,U=VUID,AC=VPID,PA=VUPW.
.ENDIF,SAMEPID.
*PURGE,PROG_L_PROBLEMID.
*MFTAKE,PROG_L_PROBLEMID.
*DEFINE,PROG_L_PROBLEMID.
COMMENT.  SEND FINAL JOB TO VSOS VIA MFQUEUE
MFQUEUE,PROG_L_JOB,ST=CYV,DC=IN.
ENQUIRE,SR.
COMMENT.  APPEND DAYFILE AND OUTPUT TO LOG FILE
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
APPEND,LOG_PROBLEMID,DF/NA.
EXIT.
COMMENT.  RETRIEVE DAYFILE AND OUTPUT ON ABORT
SAVE,SAILOUT/NA.
ENQUIRE,SR.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
APPEND,LOG_PROBLEMID,DF/NA.
.EOR
.*         **************************
.*         * BEGIN PHASE 4 ON VSOS *
.*         **************************
```

```
PROG_LRUN,STCYV.  <<<THE MAIN JOB - RUN PROG_L ON VSOS>>>
.IF,$VUID$=$SAME$.OR.$VUPW$=$SAME$.OR.$VPID$=$SAME$,SAMEPID.
USER,U=NUID,AC=NPID,PA=NUPW.
.ELSE,SAMEPID.
USER,U=VUID,AC=VPID,PA=VUPW.
.ENDIF,SAMEPID.
RESOURCE,JCAT=DEFERRED,TL=TIMELIMIT,LP=14,WS=1792.
SET,DAYFILE=OFF.
PATTACH,HULLPOOL.
ATTACH,DAT_PROBLEMID.
ATTACH,PROG_L_PROBLEMID.
COPY,DAT_PROBLEMID,TAPE5.
FTN200,I=PROG_L_PROBLEMID,B=PROG_LBIN,SC=1,LIST=0,C64,OPT=0,ERRORS=ERRLIST.
LOAD,PROG_LBIN,CN=PROG_LEXE/5000,CDF=5000,L=NULL.
PURGE,TAPE4,TAPE6,TAPE9.
.IF,$PROG$.EQ.$KEE$,DEFRTW.
COMMENT.  DEFINE TAPES 4 AND 9 AS RT=W FOR KEEL RUN
REQUEST,TAPE4,RT=W.
REQUEST,TAPE9,RT=W.
.ENDIF,DEFRTW.
DEFINE,TAPE4.
DEFINE,TAPE6.
DEFINE,TAPE9.
.IF,$PROG$.EQ.$HUL$,HFILES.
COMMENT.  ATTACH PREVIOUS KEEL FILES FOR HULL RUN
ATTACH,KEE4_PROBLEMID,AC=WRAXM.
ATTACH,KEE9_PROBLEMID,AC=WRAXM.
COPY,KEE4_PROBLEMID,TAPE4.
COPY,KEE9_PROBLEMID,TAPE9.
.ENDIF,HFILES.
.IF,$PROG$.EQ.$PUL$,PFILES.
.*
.*  WARNING - THE FOLLOWING FILE ASSIGNMENTS MAY NEED ADJUSTMENT
.*  FOR SOME USES OF PROGRAM PULL.
.*
COMMENT.  ATTACH PREVIOUS HULL FILES FOR PULL RUN
ATTACH,HUL4_PROBLEMID,AC=WRAXM.
ATTACH,HUL9_PROBLEMID,AC=WRAXM.
COPY,HUL4_PROBLEMID,TAPE4.
COPY,HUL9_PROBLEMID,TAPE9.
DEFINE,TAPE10.
.ENDIF,PFILES.
PROG_LEXE.
COMMENT.  SWITCH FILE NAMES ON SUCCESSFUL COMPLETION OF RUN
PURGE,PROG_4_PROBLEMID,PROG_9_PROBLEMID.
SWITCH,TAPE4,PROG_4_PROBLEMID.
SWITCH,TAPE9,PROG_9_PROBLEMID.
REWIND,TAPE6.
COMMENT.  SEND PROG_L OUTPUT TO NOS
COPY,TAPE6,T6TEMP.
MFLINK,T6TEMP,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
 "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
  "PURGE,PROG_L_OUT/NA.","DEFINE,PROG_L_OUT.".
PURGE,TAPE6.
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
COMMENT.  APPEND DAYFILE ETC TO LOG FILE ON NOS
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
 "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
 "APPEND,LOG_PROBLEMID/UN=NUID,NA.".
EXIT.
COMMENT.  RETRIEVE DAYFILE ETC ON ABORT
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,NUID,NUPW.","CHARGE,GROUP,NPID.",
.IF,$GROUP$.EQ.$DFC$,RFSGET.
 "PACKNAM,PN=RCYN002.",
.ENDIF,RFSGET.
 "APPEND,LOG_PROBLEMID/UN=NUID,NA.".
.EOR
```

APPENDIX B

A TYPICAL HULLJCL INPUT AND RESULTANT HULLJOB OUTPUT

```
HULLJCL INPUT:
=============
HULLJCL, NUID=DFCDLS, NUPW=NOSPASS, NPID=DFCPRF, GROUP=DFC,
VUID=MABDDR, VUPW=VSOSPAS, VPID=MABDDR, PROG=KEEL, TIMELIMIT=300,
PROBLEMID=TEST, DATFILE=MYDATA, DEFERNOS=YES.
HULLJOB OUTPUT:
==============
SUBKEEL,STDEF.  <<<SEND THE DATA TO VSOS AND START THE RUN>>>
USER,DFCDLS,NOSPASS.
CHARGE,DFC,DFCPRF.
SETJOB,DC=NO.
COMMENT.  GET OR ATTACH THE INPUT DATA TO GO TO VSOS.
GET,INDATA=MYDATA/NA.
IFE,.NOT.FILE(INDATA,AS),DOATT.
ATTACH,INDATA=MYDATA.
ENDIF,DOATT.
COMMENT.  SEND THE DATA FILE TO VSOS VIA MFLINK
MFLINK,INDATA,ST=CYV,DD=C6.
*USER,U=MABDDR,AC=MABDDR,PA=VSOSPAS.
*PURGE,DATTEST.
*MFTAKE,DATTEST.
*DEFINE,DATTEST.
COMMENT.  SEND THE REMAINING JOBS TO VSOS VIA MFQUEUE
COPYCR,INPUT,VJOB1,3.
MFQUEUE,VJOB1,ST=CYV,DC=IN.
ENQUIRE,SR.
COMMENT.  START THE LOG FILE ON NOS
PURGE,LOGTEST/NA.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
SAVE,DF=LOGTEST.
PERMIT,LOGTEST,MABDDR=A.
EXIT.
COMMENT.  RETRIEVE DAYFILE AND OUTPUT ON ABORT
ENQUIRE,SR.
PURGE,LOGTEST/NA.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
SAVE,DF=LOGTEST.
*EOR
PLANKRUN,STCYV.  <<<RUN PLANK ON VSOS TO GENERATE INPUT2>>>
USER,U=MABDDR,AC=MABDDR,PA=VSOSPAS.
RESOURCE,JCAT=DEFERRED,TL=300,LP=14,WS=1792.
SET,DAYFILE=OFF.
PATTACH,HULLPOOL.
ATTACH,DATTEST.
COPY,DATTEST,TAPE5.
FTN200,I=PLANK121,B=PLANKBIN/500,SC=1,LIST=0,C64,OPT=0,ERRORS=ERRLIST.
```

```
LOAD,PLANKBIN,CN=PLANKEXE/5000,CDF=5000,L=NULL.
PURGE,TAPE6.
DEFINE,TAPE6.
PLANKEXE.
REWIND,TAPE6.
COMMENT.  SEND PLANK OUTPUT TO NOS
COPY,TAPE6,T6TEMP.
MFLINK(T6TEMP,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
   "PACKNAM,PN=RCYN002.",
    "PURGE,PLNKOUT/NA.","SAVE,PLNKOUT/NA.")
COMMENT.  SEND SECONDARY INPUT FILE TO NOS
MFLINK(INPUT2,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
   "PACKNAM,PN=RCYN002.",
    "PURGE,IN2TEST/NA.","SAVE,IN2TEST.")
COMMENT.  SEND REMAINING JOBS TO NOS VIA MFQUEUE
COPY,INPUT,SAILJOB.
COPYL,INPUT,SAILJOB,PART=G,NUM=1,OSKIP=*.
MFQUEUE,SAILJOB,ST=CYN,DD=C6,DC=IX.
PURGE,TAPE6.
SUMMARY.
COMMENT.  APPEND DAYFILE ETC TO LOG FILE ON NOS
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
   "PACKNAM,PN=RCYN002.",
   "APPEND,LOGTEST/UN=DFCDLS,NA.".
EXIT.
COMMENT.  RETRIEVE DAYFILE ETC ON ABORT
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.".
   "PACKNAM,PN=RCYN002.",
   "APPEND,LOGTEST/UN=DFCDLS,NA.".
*EOR
SAILRUN,STDEF.  <<<RUN SAIL ON NOS TO GENERATE VERSION OF KEEL>>>
USER,DFCDLS,NOSPASS.
CHARGE,DFC,DFCPRF.
SETTL,200.
```

```
SETJOB,DC=NO.
COMMENT. RUN SAIL ON NOS TO GENERATE THE MAIN PROGRAM.
PURGE,SAILOUT/NA.
PURGE,KEETEST/NA.
DEFINE,KEETEST.
ATTACH,HVER121/UN=DFCTJW,PN=RCYN002.
ATTACH,BINLIB/UN=DFCDLS,PN=RCYN002.
GTR,BINLIB,LGO.REL/*
GET,IN2TEST.
COPYCR,INPUT,KEELJOB,1.
REWIND,*.
LGO,INPUT2=IN2TEST,OUTPUT=SAILOUT,OLD=HVER121,SOURCE=KEETEST.
SAVE,SAILOUT/NA.
COMMENT.  SEND THE PROGRAM TO VSOS.
REWIND,*.
COPY,KEETEST,LOCALF.
MFLINK,LOCALF,ST=CYV,DD=C6.
*USER,U=MABDDR,AC=MABDDR,PA=VSOSPAS.
*PURGE,KEELTEST.
*MFTAKE,KEELTEST.
*DEFINE,KEELTEST.
COMMENT.  SEND FINAL JOB TO VSOS VIA MFQUEUE
MFQUEUE,KEELJOB,ST=CYV,DC=IN.
ENQUIRE,SR.
COMMENT.  APPEND DAYFILE AND OUTPUT TO LOG FILE
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
APPEND,LOGTEST,DF/NA.
EXIT.
COMMENT.  RETRIEVE DAYFILE AND OUTPUT ON ABORT
SAVE,SAILOUT/NA.
ENQUIRE,SR.
DAYFILE,DF,PD=6.
REWIND,OUTPUT.
COPY,OUTPUT,DF.
PACK,DF.
APPEND,LOGTEST,DF/NA.
*EOR
KEELRUN,STCYV.  <<<THE MAIN JOB - RUN KEEL ON VSOS>>>
USER,U=MABDDR,AC=MABDDR,PA=VSOSPAS.
RESOURCE,JCAT=DEFERRED,TL=300,LP=14,WS=1792.
SET,DAYFILE=OFF.
PATTACH,HULLPOOL.
ATTACH,DATTEST.
ATTACH,KEELTEST.
COPY,DATTEST,TAPE5.
FTN200,I=KEELTEST,B=KEELBIN,SC=1,LIST=0,C64,OPT=0,ERRORS=ERRLIST.
LOAD,KEELBIN,CN=KEELEXE/5000,CDF=5000,L=NULL.
PURGE,TAPE4,TAPE6,TAPE9.
COMMENT.  DEFINE TAPES 4 AND 9 AS RT=W FOR KEEL RUN
```

```
REQUEST,TAPE4,RT=W.
REQUEST,TAPE9,RT=W.
DEFINE,TAPE4.
DEFINE,TAPE6.
DEFINE,TAPE9.
KEELEXE.
COMMENT.  SWITCH FILE NAMES ON SUCCESSFUL COMPLETION OF RUN
PURGE,KEE4TEST,KEE9TEST.
SWITCH,TAPE4,KEE4TEST.
SWITCH,TAPE9,KEE9TEST.
REWIND,TAPE6.
COMMENT.  SEND KEEL OUTPUT TO NOS
COPY,TAPE6,T6TEMP.
MFLINK,T6TEMP,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
  "PACKNAM,PN=RCYN002.",
   "PURGE,KEELOUT/NA.","DEFINE,KEELOUT.".
PURGE,TAPE6.
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
COMMENT.  APPEND DAYFILE ETC TO LOG FILE ON NOS
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
  "PACKNAM,PN=RCYN002.",
  "APPEND,LOGTEST/UN=DFCDLS,NA.".
EXIT.
COMMENT.  RETRIEVE DAYFILE ETC ON ABORT
PATTACH,PROB.
AUD(AUDLIST,UNIT10=AUDLIST)
SUMMARY.
TV,8+.
COPY,ERRLIST,FOUT.
TV,4+.
COPYL,AUDLIST,FOUT,PART=G,NUM=1,OSKIP=*.
DAYFILE,DF.
COPYL,DF,FOUT,PART=G,NUM=1,OSKIP=*.
MFLINK,FOUT,ST=CYN,DD=C6,
JCS="USER,DFCDLS,NOSPASS.","CHARGE,DFC,DFCPRF.",
  "PACKNAM,PN=RCYN002.",
  "APPEND,LOGTEST/UN=DFCDLS,NA.".
*EOR
```

FIGURE 2 Flowchart of a job to generate and run one of the main HULL programs